A Model of Certificate Revocation

David A. Cooper
Computer Security Division
National Institute of Standards and Technology
Gaithersburg, MD 20899
david.cooper@nist.gov

August 17, 1998

1 Introduction

Public key infrastructures (PKIs) are being fielded in increasing size and numbers, but our operational experience to date has been limited to a relatively small number of environments. As a result, there are still many unanswered questions about the ways in which PKIs will be organized and operated in large scale systems. Some of these questions involve the ways in which individual certification authorities (CAs) will be interconnected. Others involve the ways in which revocation information will be distributed. In a 1994 report, the MITRE Corporation suggested that the distribution of revocation information has the potential to be the most costly aspect of running a large scale PKI [1].

In the MITRE report, it was assumed that each CA would periodically issue a CRL that listed all of the unexpired certificates that it had revoked. Since the MITRE report was published, several alternative revocation distribution mechanisms have been proposed. Each of these mechanisms has its own relative advantages and disadvantages in comparison to the other schemes. We are working to create mathematical models of some of the proposed revocation distribution mechanisms, which we will use in order to determine under what circumstances each of the mechanisms is most efficient.

Most of the proposals have involved variations of the original CRL scheme. Examples include the use of segmented CRLs, Δ-CRLs, and on-line certificate status protocols (i.e., the on-demand issuance of an individual CRL). However, some schemes do not involve the use of any type of CRL (e.g., hash chains [3]).

In this paper, we will present a model for segmented CRLs along with some conclusions about segmented CRLs that can be drawn from the model. In comparing various options, we will assume that relying parties request revocation information only when needed to perform a validation (i.e., no pre-caching of CRLs) and that they have perfect caches (i.e., no CRL segments are deleted from the cache until they have expired). For each option, we will compute the request rate for CRL segments as a function of time. As we will show, request rates vary over time. However, in building a repository for a PKI, it is necessary to build one that is capable of handling incoming requests, even when the request rate is at its peak, without unreasonable response times. As such, we will be looking for distribution schemes that minimize peak loads on repositories as opposed to schemes that minimize average loads.

# certificates	number of certificates that have been issued by this CA
# relying parties	number of users who attempt to validate certificates signed by this CA
validation rate	average number of certificates/day that a relying party attempts to validate
revocation rate	fraction of certificates that are revoked per day
certificate lifetime	number of days between a certificates issuance and expiration
CRL update rate	number of times per day that CRLs are issued
# segments	number of CRL segments that are issued
CRL header	the cost of downloading an empty CRL
CRL entry	the incremental download cost added by each CRL entry
# entries	the average number of entries per CRL segment

Table 1: model parameters

2 The Basic Model

In the traditional model for CRLs, the certification authority periodically issues a CRL, which it posts to a repository. Any relying party requiring revocation information retrieves the CRL from the repository. In order to enhance performance, copies of the CRL may be distributed to several sites. In our model, however, we will assume that all relying parties obtain CRL information from the same repository[1].

In order to examine the load on the repository from CRL requests, we must determine the rate at which requests for CRLs are made over time. In order to do this, we will assume that CRLs are issued on a regular basis and that every relying party will require the most up to date CRL in order to validate certificates. Suppose, for example, that a new CRL is published at time 0. Then from time 0 forward (or until the next CRL is published), relying parties will need this CRL in order to validate certificates. Since we assume that each relying party has a perfect cache, each relying party will only need to download the CRL from the cache the first time, after time 0, that it attempts to validate a certificate.

In order to compute the overall CRL request rate, we need to know the probability density function for validation attempts for a single relying party. If the number of relying parties is reasonably large, then we can assume that the times at which validation attempts are made are independent of each other (i.e., they occur at randomly distributed times). We can then use an exponential interarrival probability density to model the timing of validation attempts. Then, by definition, the probability that a relying party's first validation attempt will occur in the interval $[t \ldots t + dt]$, in the limit $dt \to 0$, is

$$validation\ rate \times e^{-validation\ rate \times t} dt \tag{1}$$

Since each relying party downloads the CRL at the time of its first validation attempt after time 0, equation (1) also represents the probability that any given relying party will send a request to the repository for the CRL in the interval $[t \ldots t + dt]$. Multiplying equation (1) by the number of relying parties gives us the total expected number of requests to be made in the interval $[t \ldots t + dt]$:

[1] If more than one repository is used, then the load on each repository could be approximated by dividing the number of relying parties by the number of repositories.

$$N(t) = \#\ relying\ parties \times validation\ rate \times e^{-validation\ rate \times t} dt \tag{2}$$

Dividing both sides of equation (2) by dt gives us the request rate at time t:

$$R(t) = \frac{N(t)}{dt} = \#\ relying\ parties \times validation\ rate \times e^{-validation\ rate \times t} \tag{3}$$

Given that each CRL is valid for $\frac{1}{CRL\ update\ rate}$ days, the CRL request rate at any time can be computed as $R\left(t \bmod \frac{1}{CRL\ update\ rate}\right)$.

3 Segmented CRLs

We will now generalize the equations from the previous section to the case in which revocation information is divided among some number of CRL segments. In some cases, certificate revocation information may be divided among CRL segments in a way that attempts to minimize the number of CRL segments that a relying party will need to download. In our model, however, we will assume that revocation information is distributed at random among the CRL segments. From this assumption, we can conclude that each validation attempt is equally likely to require access to any of the CRL segments.

As in the previous section, we will begin by determining the probability density function for a single relying party with respect to a single CRL segment (segment 1). A relying party will request segment 1 from the repository in the interval $[t \ldots t + dt]$ iff it attempts to validate a certificate in the interval $[t \ldots t + dt]$ that requires the use of segment 1 and it has not validated any certificates in the interval $[0 \ldots t]$ that required the use of segment 1.

We will first determine the probability that a relying party will not have requested segment 1 in the interval $[0 \ldots t]$. Since we are assuming an exponential interarrival probability for validation attempts, we know from the Poisson law [5] that the probability that n validation attempts will be made during an interval of length t is

$$\left[\frac{(validation\ rate \times t)^n}{n!}\right] e^{-validation\ rate \times t} \tag{4}$$

Since there is a probability of $\frac{1}{\#\ segments}$ that segment 1 will be needed to perform any given validation attempt, the probability that segment 1 will not be needed for any of n validation attempts is

$$\left(1 - \frac{1}{\#\ segments}\right)^n \tag{5}$$

Combining equations (4) and (5), we see that the probability that any given relying party will not request segment 1 during the interval $[0 \ldots t]$ is

$$\sum_{n=0}^{\infty}\left(1-\frac{1}{\#\ segments}\right)^{n}\left[\frac{(validation\ rate\times t)^{n}}{n!}\right]e^{-validation\ rate\times t} \tag{6}$$

which can be re-written as

$$e^{-validation\ rate\times t}\sum_{n=0}^{\infty}\frac{1}{n!}\left[\left(1-\frac{1}{\#\ segments}\right)(validation\ rate\times t)\right]^{n} \tag{7}$$

From Taylor's Theorem [6] we know that

$$e^{x}=\sum_{n=0}^{\infty}\frac{x^{n}}{n!} \tag{8}$$

So, equation (7) can be simplified to

$$e^{-validation\ rate\times t}e^{(1-1/\#\ segments)(validation\ rate\times t)}=e^{-(validation\ rate\times t)/(\#\ segments)} \tag{9}$$

Next, we need to determine the probability that a relying party will need segment 1 during the interval $[t\ldots t+dt]$ (in the limit $dt\to 0$). The probability that one validation attempt will be made in the interval $[t\ldots t+dt]$ is[2]

$$validation\ rate\times e^{-validation\ rate\times dt}dt \tag{10}$$

Since

$$\lim_{dt\to 0}e^{-validation\ rate\times dt}=1 \tag{11}$$

equation (10) can be simplified to $validation\ rate\times dt$. Since the probability that any given validation attempt will require the use of segment 1 is $\frac{1}{\#\ segments}$, the probability that segment 1 will be needed in the interval $[t\ldots t+dt]$ is

$$\frac{validation\ rate}{\#\ segments}dt \tag{12}$$

By combining equations (9) and (12) and by multiplying the result by the number of relying parties, we get that the total expected number of requests for segment 1 in the interval $[t\ldots t+dt]$ is

$$N_{s}'(t)=\frac{\#\ relying\ parties\times validation\ rate\times e^{-(validation\ rate\times t)/(\#\ segments)}dt}{\#\ segments} \tag{13}$$

[2]Since the interval $[t\ldots t+dt]$ is infinitesimally small, we can assume that the probability of more than one validation attempt occurring is 0.

Dividing both sides of equation (13) by dt gives us the request rate for segment 1 at time t:

$$R'_s(t) = \frac{N'_s(t)}{dt} = \frac{\#\ relying\ parties \times validation\ rate \times e^{-(validation\ rate \times t)/(\#\ segments)}}{\#\ segments} \tag{14}$$

Finally, if equation (14) is multiplied by the number of segments we get the total request rate:

$$R_s(t) = \#\ relying\ parties \times validation\ rate \times e^{-(validation\ rate \times t)/(\#\ segments)} \tag{15}$$

Equation (15) shows us how CRL request rates change with the amount of segmentation. Since $R_s(0) = \#\ relying\ parties \times validation\ rate$, it is clear that the peak validation rate is not affected by the amount of segmentation. This would suggest that increasing the amount of segmentation will not lead to the need for more powerful repositories. In fact, if repositories can, in general, service requests for shorter CRL segments faster than requests for longer CRL segments, then this model suggests that increasing the amount of segmentation would allow for the use less powerful repositories to handle requests for CRL segments.

While not the focus of this paper, we must acknowledge that there are some drawbacks to segmentation. First, while the peak request rate is not affected by segmentation, the average request rate increases with the number of segments used since the request rate drops off more slowly. This difference is depicted in figures 1 and 2. Both figures show CRL request rates over the course of 24 hours for scenarios in which CRL information is updated once a day (at time 0 in these graphs) and in which there are 300,000 relying parties each validating an average of 10 certificates per day. Figure 1 represents the case in which there is a single CRL and figure 2 represents the case in which revocation information is divided between two CRL segments. The increased average request rate may not be a problem if the repository is dedicated to distributing CRL information. However, if the repository is also used to distribute other information, then it may be preferable to minimize the total number of requests rather than just the peak rate.

Segmented CRLs may also be less preferable from the relying party's point of view. Each time that a relying party needs a CRL segment that is not already in its cache to perform a validation it must send a request to a repository and wait for the response before it can complete the validation. As the number of CRL segments increases, so does the number of times that the relying party will have to wait for CRL information from the repository.

4 Staggered CRL Issuance

In section 3 we assumed that all CRL segments were issued at the same time. However, given that the request rate for a CRL segment declines over time, it may make more sense to stagger the issuance of CRL segments. Taking the example from figure 2, we see that the request rate for each segment at the time of issuance is 17.36 requests/second. However, after 12 hours, the request rate has dropped to 1.43 requests/second. So, if the issuance of the two CRL segments were staggered by 12 hours then the peak request rate would be only 18.79 requests/second as opposed to a peak rate of 34.72 requests/second if both segments were issued at the same time (see figure 3). If there were 3 CRL segments issued at 8 hour intervals then the peak request rate would drop to 16.64 requests/second (see figure 4). Unfortunately, the peak request rate does not continue to decline

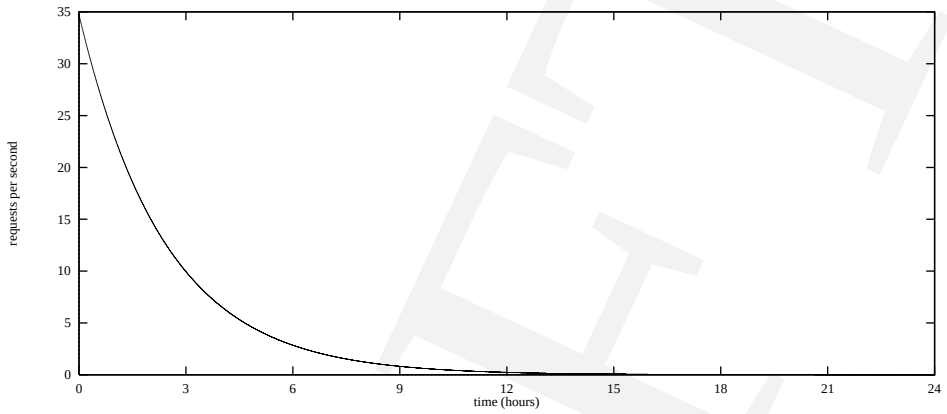

Figure 1: Unsegmented CRL

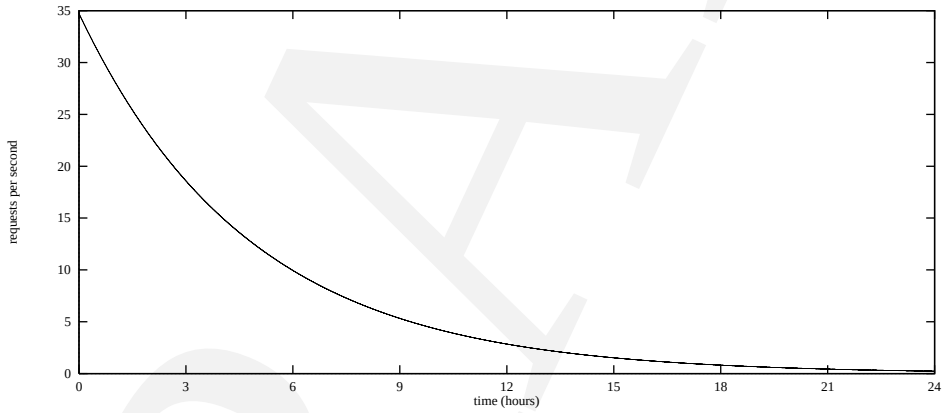

Figure 2: Two CRL Segments

with increasing numbers of segments. With 4 CRL segments issued at 6 hour intervals, the peak request rate increases slightly to 17.15. As the number of CRL segments approaches infinity, the peak request rate approaches the peak rate for an unsegmented CRL (see figure 5).

In general, the request rate for segmented CRLs issued at evenly separated intervals is:

$$\frac{\#\ relying\ parties \times validation\ rate}{\#\ segments} e^{-(validation\ rate \times t')/(\#\ segments)} \sum_{i=0}^{s-1} e^{-(i \times validation\ rate)/\left(CRL\ update\ rate \times (\#\ segments)^2\right)} \tag{16}$$

where $t' = t \bmod \left(\frac{1}{CRL\ update\ rate \times \#\ segments}\right)$

Of course, in order to determine the optimal number of segments to use, one must consider service rates in addition to request rates. Once both the request and service rates are known, a queuing system can be used to determine the mean waiting times for relying parties for various scenarios. Since the request rate is not constant, the equations to determine the exact mean waiting times are rather complicated (see [2, 4]). However, the steady state solution for constant request and service rates appears to provide a good approximation as long as the service rate exceeds the

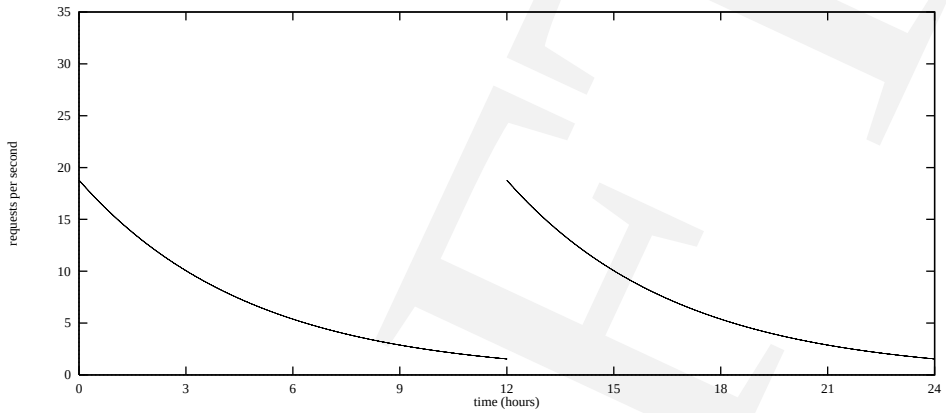

Figure 3: Two CRL Segments with Staggered Issuance

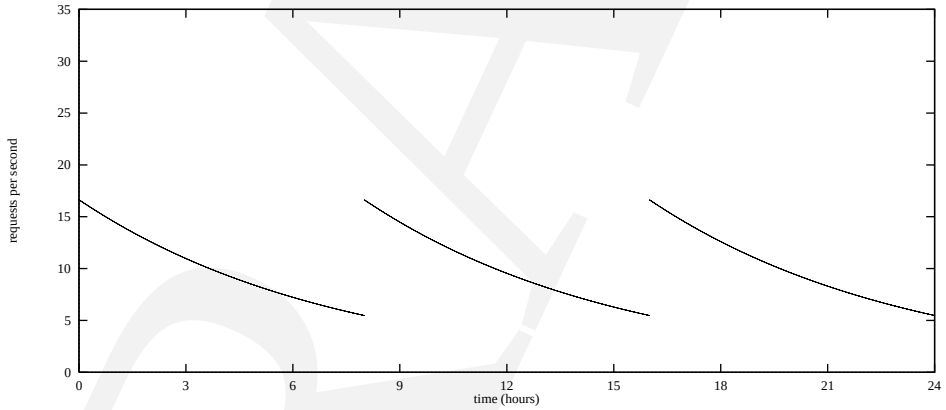

Figure 4: Three CRL Segments with Staggered Issuance

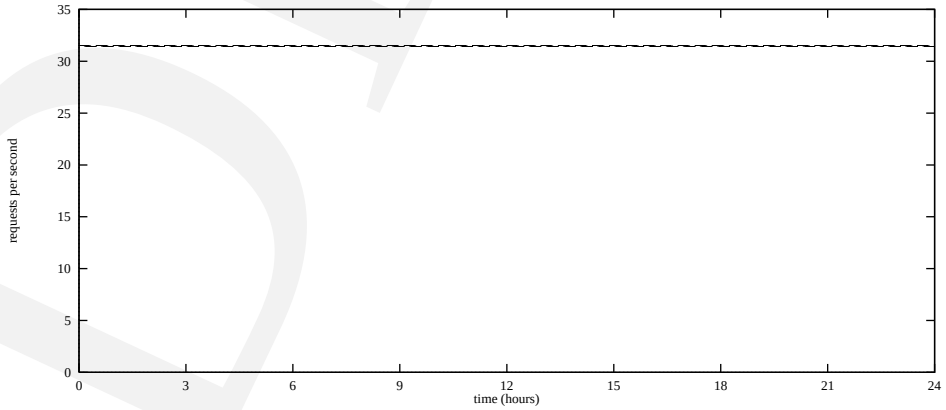

Figure 5: Fifty CRL Segments with Staggered Issuance

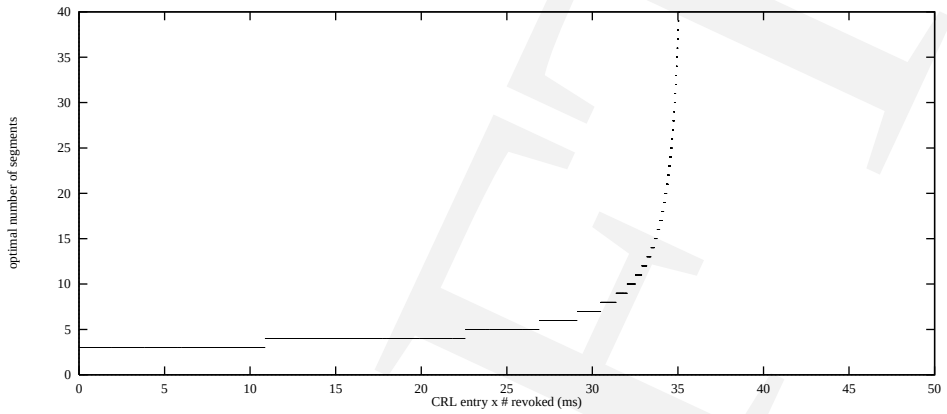

Figure 6: Optimal Number of Segments with CRL Information Updated Daily

request rate by a sufficient amount. Using this approximation, we get that the mean waiting time is $WT \approx \frac{1}{\mu - \lambda(t)}$ where $\lambda(t)$ is the request rate and μ is the service rate [5].

If we assume that the amount of time needed to service a request for a CRL segment grows linearly with the size of the CRL segment, then average service time can be written as $ST(s) = \mathit{CRL\ header} + \mathit{CRL\ entry} \times \#\ \mathit{entries}$. If $\#\ \mathit{revoked}$ is the average number of unexpired certificates that have been revoked then $\#\ \mathit{entries} \approx \frac{\#\ \mathit{revoked}}{\#\ \mathit{segments}}$ and so $ST(s) \approx \mathit{CRL\ header} + \frac{\mathit{CRL\ entry} \times \#\ \mathit{revoked}}{\#\ \mathit{segments}}$.

By using equation (16) to compute $\lambda(0)$ and by using $\frac{1}{ST(s)}$ as an estimate for μ, we can determine, for any given environment, the optimal segmentation scheme to use in order to minimize the worst case mean waiting time for relying parties (or, alternatively, the scheme that will allow for the use of the least powerful repository).

Figure 6 presents one example of how the optimal number of segments changes as the service time becomes more dependent on the size of the CRL segment that is being requested. In this figure, as before, we are using a scenario in which there are 300,000 relying parties each attempting to validate 10 certificates per day. In addition, revocation information is updated daily and the issuance of CRL segments is evenly spread throughout the day. For all points on the graph, values for *CRL header* and $\mathit{CRL\ entry} \times \#\ \mathit{revoked}$ were chosen so that $\mathit{CRL\ header} + \mathit{CRL\ entry} \times \#\ \mathit{revoked} = 50\,\text{ms}$ (i.e., $\mu = 20\,\text{services/second}$ for unsegmented CRLs). As predicted earlier, when the service rate is independent of the amount of segmentation, breaking CRLs into 3 segments is optimal. At the other extreme, when the mean service time is directly proportional to the CRL segment size (i.e., no fixed cost), mean waiting time is minimized by using individual CRLs.

Figure 7 shows the optimal segmentation values for a scenario in which revocation information is updated every 10 minutes. Just as in the scenario used above, there are 300,000 relying parties each attempting to validate 10 certificates per day. However, in order to handle the increased demand for revocation information, we needed to increase the service rate to $\mu = 40\,\text{services/second}$. So, in this figure, $\mathit{CRL\ header} + \mathit{CRL\ entry} \times \#\ \mathit{revoked} = 25\,\text{ms}$. Since, in this scenario, there is only a 0.23% chance that any given relying party will attempt to validate more than one certificate in a span of 10 minutes, we can expect that almost every validation attempt will require downloading a new CRL segment from the repository, even if unsegmented CRLs are used. As a result, there is little benefit to downloading and caching large CRLs. This can be seen in figure 7, which shows

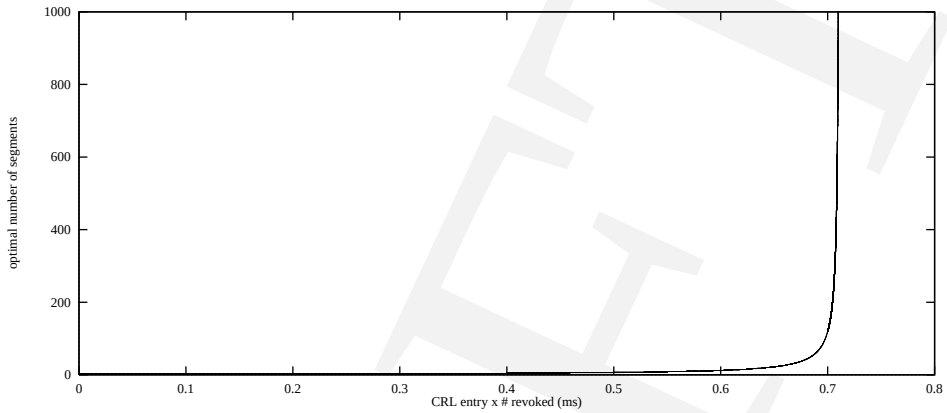

Figure 7: Optimal Number of Segments with CRL Information Updated Every 10 Minutes

that it is only optimal to use a relatively small number of segments if the service rate is unaffected by the segment size. Once more than 0.7 ms of the 25 ms service time for unsegmented CRLs is based on the CRL size, it is best to use individual CRLs.

There are some conclusions that we can draw from figures 6 and 7. As the number of revoked certificates increases, we can expect the cost of downloading CRL information to increase. Since the fixed cost (i.e., the cost of downloading an empty CRL segment) will not change, increasing the number of revoked certificates will move us further to the right on the x axes of the two charts. In other words, more segmentation is better when high revocation rates are expected and less segmentation is better when low revocation rates are expected.

Larger CRL segments (i.e., less segmentation) also appear to be preferable when the revocation information is updated relatively infrequently. The longer that a CRL segment is valid, the more likely that a relying party will use that segment more than once. In other words, caching large CRL segments will reduce the request rate to the repository more if the segments are valid for a longer period of time. This can be seen in figures 6 and 7 where the optimal number of segments for revocation information updated daily is always significantly smaller than the optimal number for revocation information updated every 10 minutes.

References

[1] Shimshon Berkovits, Santosh Chokhani, Judith A. Furlong, Jisoo A. Geiter, and Jonathan C. Guild. *Public Key Infrastructure Study: Final Report*. Produced by the MITRE Corporation for NIST, April 1994.

[2] George Luchak. The solution of the single-channel queuing equations characterized by a time-dependent poisson-distributed arrival rate and a general class of holding times. *Operations Reseach*, 4(6):711–732, December 1956.

[3] Silvio Micali. Efficient certificate revocation. Technical Memo MIT/LCS/TM-542b, Massachusetts Institute of Technology, Laboratory for Computer Science, March 1996.

[4] H. M. Srivastava and B. R. K. Kashyap. *Special Functions in Queuing Theory And Related Stochastic Processes*. Academic Press, Inc., 1982.

[5] Andrew S. Tanenbaum. *Computer Networks*. Prentice-Hall, Inc., second edition, 1989.

[6] George B. Thomas, Jr. and Ross L. Finney. *Calculus and Analytic Geometry*. Addison-Wesley Publishing Company, sixth edition, 1984.